Local Geography Educator

Martin Sellergren
Uppsala university
Summer 2018
…...

Abstract

100-200 words

Contents

1 Introduction

Since recent years there has aroused a new popular way of education, through the mobile platform. Many mobile applications exist with an educational purpose, and some has grown very popular. It turns out the mobile platform is well suited for these purposes. A mobile phone it's easily accessible, it's often with you and it's easy to do a little studying whenever you feel like it and have some time to spare. Moreover, mobile application developers have good potential to make an application stimulating and entertaining using this platform.

The intent of this project is to produce a mobile application capable of teaching local geography to its users. The meaning of local geography in this context is villages, city-districts, roads, parks, rivers, schools and such geographic objects - object inside a limited area, like a city. The goal is to make a stimulating, entertaining and above all educational mobile application. This application will be referred to as *Location Educator* in this document.

Location Educator can for example be useful for someone who wants to gain better knowledge of the city where they live or where they want to go to. People who have a desire to be more knowledgeable about some place in the world. There might also be uses in professional areas. People working with transportation or public communication might benefit greatly. For example taxi and bus drivers. Not mainly to find the way to where they are going, which they likely use a GPS for, but more so for being able to answer questions from the customers about directions and distances.

Several mobile applications for geographic education currently exist. Some examples are *Seterra*, *World Geography*, *Geo Challenge* and *MapPie*. These are all based on education through quizzes. You are ranked based on the result of a quiz and the goal can be to complete many quizzes with good rankings, get a new high-score or progress to new levels. Such a design enables learning through entertainment and will also be integrated into Location Educator. A defining difference with other applications is this project's focus on local geography, where others usually focus on a global or regional scale, such as continents, oceans, states, provinces and cities.

2 Background

To satisfy the project goal of making a stimulating, entertaining and educational mobile application, careful consideration needs to be taken about the application's design. In this section, relevant areas are investigated and steps towards the design decisions are described. Next section describes the derived design.

2.1 Targeted audience

Good human-computer interaction is a delicate subject. Several aspects need to be considered when designing interfaces, especially when it comes to achieving educational value. Different people think in different ways, have different experiences and and moods. Functionality that feels natural to some might be foreign to others. Human focus is controlled by interest, our memory and patience is limited. There is always a learning curve when exposed to a new system.

To increase the chance of good human-computer interaction, it’s appropriate to define a targeted audience. The targeted audience of Location Educator is broad, it’s defined as anyone interested in learning local geography. Users might be of different ages with different experiences. Therefor, care is taken to make the interface self-explanatory and easy to interact with. Simplicity is favored over complex functionality. Focus is on inexperienced users over those preferring more advanced functionality. In order to still accommodate many different users, Location Educator will support different speeds in the education progression.

2.2 Educational aspects

It’s important to keep *human factors* in mind. How do we remember and learn? The human learning process is complex. To learn something new isn’t a trivial thing, especially not to learn something and remember it for a long time. It’s important that the person has an interest and desire to learn.

A key to learning is associations. The more associations the better. If you hear a name of a town you’ve never heard of, it’s easily forgotten. Then, you see it’s location on a map, you here a story about the magnificent mountains there and you meet an inhabitant who turns out to have very red skin, then the name has lot’s of associations, and you’ll more likely remember it. Finally you realize the name sounds a bit like *red mountains* and you’ll never forget it.

The mind forgets and needs to be reminded. Therefore is repetitions important for the learning process. The longer the knowledge has existed, the less often you need a reminder.

Stimulation is important. A healthy amount of stress will facilitate the learning process greatly. Too little stress and you start thinking of other things. To much and you’ll be disturbed and try to escape it.

Finally, it’s important to realize that errors and mistakes are okay – it’s a part of the learning process. This is something very important to keep in mind when you try to educate someone. At the time of a mistake, don’t say “Wrong -next question!”. Instead, take this opportunity to make it right, explain why it’s wrong and let the person try again, perhaps this time with a clue.

2.3 Initial potential user considerations

An interview with a potential user was performed before any major design decisions were made. The person was shown some initial sketches of a design. Through discussion, it became clear that a light workload for the user is of importance. Initially, the plan was to let a user study a particular area for a considerable amount of time and then perform an exam for this area in order to move on to next area and level. After the interview this idea was modified into a design that doesn’t involve the risk of users getting stuck in the exercise-progression as easily. The new design was discussed with the potential user who agreed that this was an improvement. The modified design involved lightweight levels with subsequent reminders (more details later in the report).

2.4 Existing application analysis

There currently exist multiple geography educational applications. They are often based on learning through quizzes. The user answers questions like "Where is Stockholm located?", or "What city is this?" and answers through tapping in a map or picking one from multiple alternatives. But the

applications differ greatly in key aspects like how to stimulate motivation and goal. Some of these applications has been studied in detail in order to identified strengths and weaknesses from an educational point of view. The results from this study is described in next subsection. Focus of this study was on the following applications: *Seterra, World Geography, Geo Sverige*, *Geo Challenge*, *Countries of the world, MapPie* and *Blank Map Quiz*.

Worth pointing out is that these applications all have a focus on global or regional geography, that is education of things like continents, oceans, countries, capitals and important landmarks. Despite this difference in focus with Location Educator, they can offer valuable insights in how to, and not to, design an educational geography application.

Furthermore, the language education application *Duolingo* was studied. This application is very popular, which I believe is mainly because of an effective education design. Duolingo also supports quiz-based education and has provided inspiration and ideas.

2.5 Key-functionality for education

Through the existing application analysis and other considerations, some key-functionality-aspects regarding educational value in a quiz-based educational application has been suggested. These aspects are described in the subsections below, along with how they are handled in applications in the existing applications study. Considerations about how successful these applications have been in these areas was used as foundation for design decision of Location Educator.

2.5.1 Motivation

Why do you want to continue using the application? Probably because you want to learn geography, but there is need of a specific motivator that shows you that you are making progress and drives you to keep going. One way to implement this is a hunt for high-score – you complete a quiz and get a score, then retakes the quiz to improve the score. The motivator is to achieve good scores on lot's of quizzes. Another motivator is level progression – unlock levels by getting good results in levels currently unlocked, and try to unlock everything (at which point you are a master in this area). Some of the studied applications use a high-score system, some use level progression and some use both.

An example of an application that is high-score based is *Seterra*, where lot's of quizzes are predefined (that is, the content of the quiz doesn't change next time you take it). The user can freely select interesting quizzes and tries to get a good high-score on those. This gives the user great freedom to select what to learn, and it's a powerful educational strategy to retake quizzes and improve the score. It’s a comfort for the user to be able to come back to the same quiz that has been taken before. A downside might be that it may seem daunting to select a quiz from the big selection. And also, there is a lack of overall progression which otherwise could provide the user with the sense of going somewhere. The great freedom is good in many ways but also means the user must be disciplined in selecting relevant quizzes and retake them before they are forgotten.

An example of an application that uses level progression is *World Geography*. Here, a quiz is automatically and semi-randomly generated (from some specifications from the user) and by completing a quiz you gain points that eventually takes you to next level. A new level means an increase in difficulty and more options for quiz-specification. This system is good for motivation

since there is a global progression (reaching new levels), and it opens up for a guided increase in difficulty.

2.5.2 Question design

The difficulty of a question should ideally be related to the user's knowledge so that it becomes challenging to answer and not too difficult or easy. If a user only has a slight idea of a subject, it's very rewarding with easy questions that capture this little knowledge and strengthen it.

Question asking and answering should be swift and simple. Questions may vary in difficulty, and it's important for initial questions to be easy to answer. It's a good idea to (to some extent) guide the user to the correct answer so the user leaves the question feeling strengthened and not confused.

Multiple choice questions are good in this regard. "What is the name of the road?", a map with a highlighted road is presented along with some answer alternatives, where the user picks one. Answering is swift and simple, and the user is gently guided to the correct answer since there is a limited number of of alternatives to pick from. The user may also be exposed to names of other geographic objects which never is a bad thing. This is a very popular question type in the studied applications.

Answer through typing is not nearly as common, but exists for example in the application *Geo Sverige*. It's often quite tedious, but opens up for a new level of difficulty which sometimes may be preferred.

Another question type is *pin-on-map*-questions where the user answers by tapping somewhere in a map. Two types exist of this, either tapping a blank map, or tapping a map with predefined answer points or areas. Tapping a blank map means the user will have to tap close enough to the exact answer. *Geo Challenge* is an example of an application using this. A problem is that the user will never be able to tap exactly correctly so I believe such questions will have a tendency to leave users feeling unsatisfied. Tapping predefined points or areas may be preferable, which also has the potential to guide the user in correct direction because of the limited number of answer alternatives. *Seterra* is an application which relies solely on this type of question.

As mentioned in the Educational aspects section, it’s important with multiple associations to support the memory during the learning process. Therefor it may be important with many different types of questions, to attack a subject from different angles. Multiple associations support learning is a fact that is beautifully utilized by the application *Countries of the world*, where questions are presented not solely through text or map. In addition, flags and capitals are included in questions.

2.5.3 Educational support

What should happen when the user answers a question incorrectly? This is a very important aspect when it comes to educational value, as mentioned in Educational aspects. An incorrect answer is a great opportunity to offer educational support.

Some of the studied applications offers no support, instead moving on to next question, leaving the user feeling confused. Most commonly, applications give the user multiple attempts with a question. The number of attempts varies, often three, sometimes more. Often, I would prefer less number of attempts. If the first or second answer is wrong it’s likely that I don’t know the answer. Then I

would prefer to be given the answer and move on instead of keep guessing. *Seterra* gives the user three attempt, then reveals the answer and waits for the user to click the correct answer. I find it pleasant to always leave a question after clicking the correct answer, even if it was revealed.

It's common to repeat incorrect questions after the quiz; giving the user another chance, which for example is the case in *World Geography* and *Duolingo*.

2.5.4 Stimulation

As described in Educational aspects section, stimulation is often a good thing during education. In the studied applications, different ways of implementing stimulation exists. Two main categories in this area exist; a clock ticking down, and a clock ticking up.

In the application *Geo Challenge*, a quiz is assigned some time and the clock is ticking down. The more questions you answer before time runs out, the better score. This provides lot's of stress, way too much according to me. *World Geography* also has a clock ticking down but it is reset after each question, and the given time is generous. This becomes a mild stimulation which provides a good amount of stress.

A clock ticking up is also common. *Seterra* uses this for example. The time taken to complete a quiz is recorded along with the score, and the goal is to minimize time and maximize score. Through this method, the amount of stress provided through the stimulator is very much dependent on the user. If the user decides to ignore the time, it offers no stimulation at all. Or the user could hunt for new time-records making the stimulator very strong.

Also, some applications don't use a stimulator at all, for example *Duolingo*.

2.5.5 Visual appearance

The final suggested key-functionality for education is simply the visual appearance of the quiz. Things like beautiful and contrasting colors, non-cluttered screens and well-thought screen-compositions are very important. Failure here could easily ruin an otherwise good application. This is in some sense the case with the application *MapPie*, where the presented quizzes are very dull in appearance. *Blank Map Quiz* is an example of the opposite. Here, the coloring of the maps presented during quizzes are unsuspected and beautiful.

2.6 Scenario analysis

When making design decisions, it helps to visualize potential users, describe them, their needs and expectations:

- Student in new city, wants to know city better. Busy with school work and activities during daytime but likes to do some light studies of the city before bedtime, for just a couple of minutes. By knowing the city better, the student feels more at home there.
- Older person thinking about moving to Portugal in wintertime. Has found a village that seems pleasant and likes to study it in detail and dream about the future. Likes the native names of little bakeries and landmarks. Has been guided into learning the application *Duolingo* and through it refresh the language. Is therefor familiar with the concept of

quizzes and progression, even though the person usually finds mobile applications hard to understand.

- A 12-year old who likes to play mobile games and compete with friends. Found Location Educator and likes it since it somehow feels meaningful in a way other games don’t.
- A taxi-driver who don’t like to take up his smart phone every time someone asks him for time estimates and directions. Also, he’s embarrassed his boss might find out his really not very knowledgeable about the city. So on his vacation, he focuses flat out on the studies.
- A sailor who plans to explore the archipelago of Stockholm. Wants to learn the names of the little islands since it helps with navigation, and helps preserving his memories of the places he passes.

2.7 Functional requirements

Functional requirements were derived, with support from previous considerations. These are requirements of specific behavior of the application.

- Construct an exercise from a size-limited area anywhere in the world.
- Maintain a list of exercises.
- Group geographic objects inside an exercise-area into categories based on characteristics.
- Group geographic objects inside an exercise-category into levels based on characteristics.
- Let user take a *level quiz* which takes the user to next level if finished with satisfactory result. The quiz has questions about the geographic objects of this level.
- At certain times, require the user to take a *reminder quiz* – a quiz with questions about geographic objects that has already appeared in a level quiz and is estimated to be weak in the memory of the user. These are sometimes required in order to open up a level quiz.
- Integrate *follow-up quizzes*, which follows another quiz with incorrect answers from that quiz.
- Include a stimulator (or stressor) during a quiz, through a clock ticking down which is reset after each question.
- Display a quiz progress-bar, as well as overall exercise-progress.
- Support three different types of quiz-questions; *Name-it*, *Place-it* and *Pair-it*.
- Associate each geographic object with a color based on characteristics. This color is revealed when the object is presented in a map or through text.
- Integrate *tapping practice*, where a user can practice outside of a quiz.
- Support predefined exercise-areas during exercise creation.
- Exercise-area-sharing and progress-sharing between users.

2.8 Non-functional requirements

Here follows requirements about what the application should *be.*

- The application is required to be educational in local geography. This involves many things; like supporting educational aspects and being graphically and functionally pleasant.
- The application should promote learning about big and important geographic objects before less significant objects.
- Swift, stimulating, lightweight quizzes with basic design, supporting *does what you think*.
- Introduce additional associations when asking questions about a geographic object.
- Educational support during a quiz, by guiding user towards correct answer when incorrect answer is given.
- Let user know that he/she is making progress towards a goal.
- Prevent user from forget any previously seen geographic objects. Prefer to remind user of those before introducing new ones.

3 Design

Here follows the intended design of Location Educator through descriptions of the different activities of the application. The images are mock-ups produced using image creating software. This design is the goal of the subsequent implementation.

3.1 Select an exercise

The *select exercise* activity displays a list of existing exercises. The user may click on an exercise to enter it, or create a new one. Exercises are reordered through drag and drop, and deleted through a swipe gesture with following confirmation. Menu button top left has various items: About, Feedback, Logout. The mock-up of this activity is presented in Figure 1.

3.2 Create a new exercise

In this activity, the the user can create a new exercise. The user specifies name and area. The area is drawn using draw-tools of the embedded map and can be located anywhere in the world. Only smaller areas are allowed (the application is for *local* geography). This restriction is enforced by disabling drawing tools if map is too zoomed out. If this is the case, the map will appear gray and a button appears that, when clicked, animates the zoom of the map to an allowed level. When zoom level is acceptable, drawing is initiated by tapping the map. As soon as the first node is drawn, zooming and scrolling is disabled. The user may clear the drawn path through a button that appears when the first node is placed. The goal with this design is to enforce the size-restriction of the specified area in a way that is self-explanatory to the user. The mock-up of this activity can be seen in Figure 2.

The functional requirements mentions predefined exercise-areas, which is one way to enable different users to take the same exercise. This may be of interest to users who for instance likes to

compare scores with others. The mock-up doesn’t include this feature as this requirements is put aside initially.

3.3 Inside an exercise

When entering an exercise you’re presented with a list of categories, each containing levels, as seen in Figure 3. There are a maximum of five categories, less if the exercise-area doesn’t contain any geographic objects of that category. These categories are as follows:

- **Settlements** – geographic objects that may be described as settlements, including cities, towns, villages and neighborhoods. This category also holds things like graveyards, boatyards, and golf courses, basically any kind of named local area that is under human care.
- **Roads** – roads of any sort. Also paths.
- **Nature** – for example forests, nature reserves, islands, beaches, mountains and water bodies of any sort. Also parks of any sort.
- **Transport** – geographic objects related to transportation, for example bus stations, train stations, airports, and ferry terminals.
- **Constructions** – buildings, bridges, tunnels, monuments and so on. Basically anything build by man.

By clicking a category, you are presented with three options: enter a tapping-exercise, attempt a level-quiz or take a category-reminder-quiz. These are described in next sections.

Top right is the overall exercise progression. Down right is a button for taking an exercise-reminder-quiz (as opposed to the more restricted category-reminder-quiz). Top left is the menu button which in addition to the items in the menu of the *Select exercise activity* has the option “Select exercise” that takes you to the Select exercise activity. This options menu may also contain functionality regarding sharing progress and exercise-area with other users.

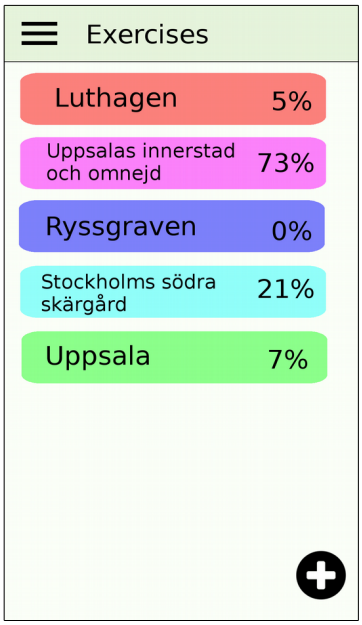

Figure 1: Mock-up of the Select exercise activity.

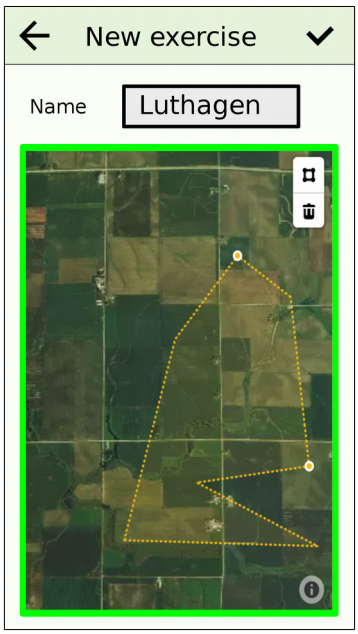

Figure 2: Mock-up of the New exercise activity.

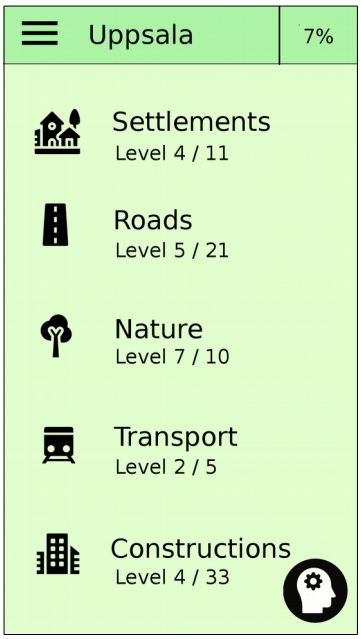

Figure 3: Mock-up of the main activity of an exercise called Uppsala.

3.4 Tapping practice

Tapping practice is useful if a user wants to try his/her knowledge outside of a quiz. It's casual practice which may be enjoyed by some users. The same or similar idea is implemented in the application *Seterra*.

The user is presented with a map containing geographic objects of a certain category in an exercise. The user may freely tap objects in order to display their names for a brief period of time. This is particularly useful when a user wants to attempt to name geographic objects and instantly receive confirmation if the guess was correct. A button above the map let's the user decide if to display geographic objects from next level or from all past levels. See the mock-up in Figure 4.

3.5 Quiz

A quiz handles around five geographic objects, with three to five questions each. There are three different types of questions, *Name-it*, *Place-it* and *Pair-it*, described in detail in next sections. A question must be answered within a certain, generously provided, amount of time and if no answer is given, the correct answer is shown (this is a stimulator). Each object is associated with a color loosely based on its characteristics, which is used when the object is presented in a question, in order to increase the number of association the user has to the object.

There are three different types of quizzes:

- **Level quiz** – for level progression. This quiz has questions about geographic objects of a certain category at a certain level. Increasing levels means less significant objects. The first level may for instance have motorways and the last pathways. Passing a quiz with satisfactory result takes you to next level of this category.

- **Reminder quiz** – for being reminded of previously seen geographic objects, from previous levels. Problematic and old (long time no see) objects are favored when selecting objects for such a quiz. Reminder quizzes exists both on category-level (for objects in a particular category) and exercise-level (for any seen objects in the exercise). You are required to take a reminder quiz (or a couple) now and then in order to continue level progression, as seen in Figure 5 and 6.
- **Follow-up quiz** – which follows an other quiz (that isn't follow-up) and repeats the incorrectly answered questions.

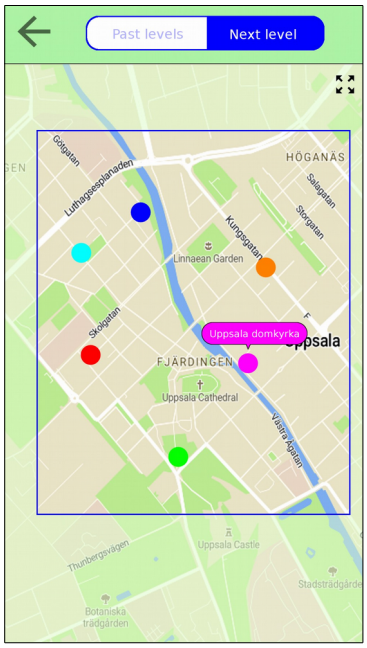

Figure 4: Mock-up of tapping practice.

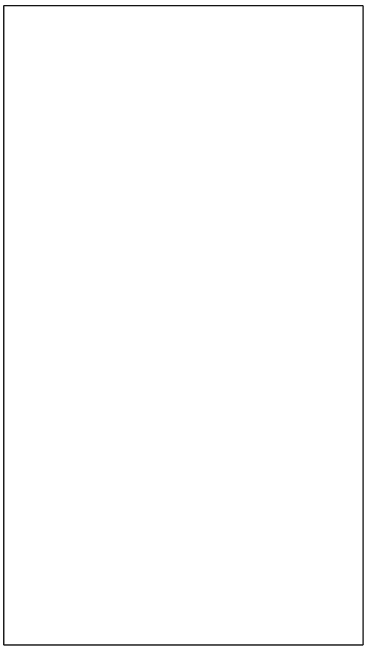

Figure 5: Mock-up of an exercise activity when an exercise-reminder is required.

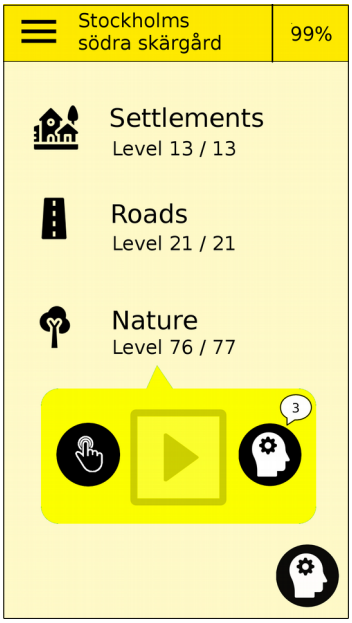

Figure 6: Mock-up of an exercise activity when a category-reminder is required.

3.5.1 Quiz question: Name it

A multiple choice question where a geographic object is presented in a map. If correct answer is given, moves on to next questions. If incorrect, the correct answer is hinted and the user chooses the correct alternative before moving on to next question. The number of alternatives is defined by the number of previous questions about this geographic object in the quiz. Initially there are two alternatives, later four and even six. A mock-up is shown in Figure 7.

3.5.2 Quiz question: Place it

In this question you are supposed to tap an object presented in a map after being given a name and a category. An object may be presented in the map as a single marker, a line or a closed line. Similar concepts regarding answering sequence and alternatives count as for Name-it questions. See the mock-up in Figure 8.

3.5.3 Quiz question: Pair it

Here you are supposed to pair name with location, by first tapping a name and then tap the corresponding object in the map. Complete all pairings before the question is done. The objects of this question might not belong to the same level or even category as the quiz. The only requirement is that the objects have been seen in this or earlier quizzes. Therefor, this question-type has the additional function of reminding the user of previously seen objects. The mock-up is presented in Figure 9.

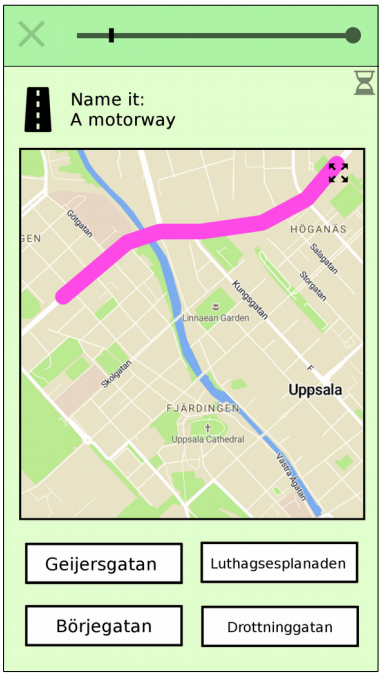

Figure 7: A Name-it question mock-up.

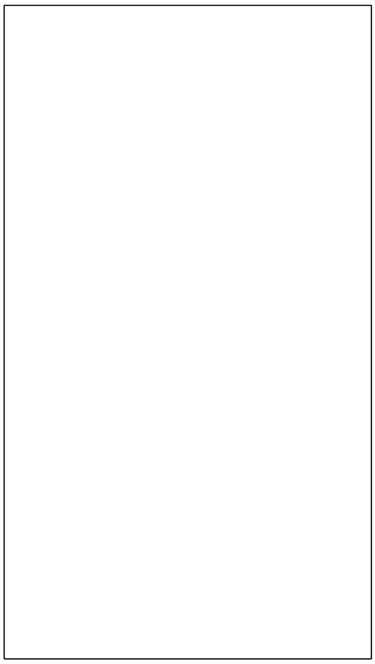

Figure 8: A Place-it question mock-up.

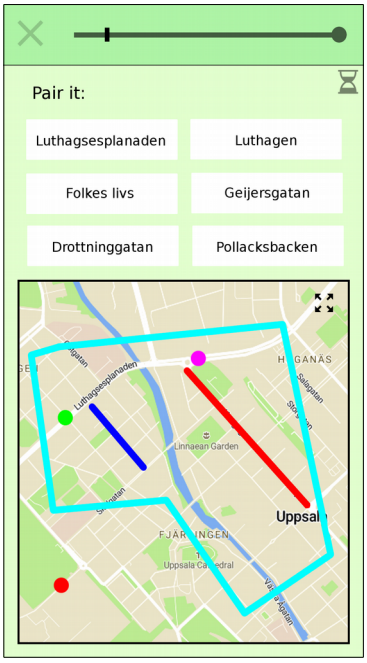

Figure 9: A Pair-it question mock-up.

4 Implementation

It was decided to implement Location Educator for the Android operating system for use by Android mobile phones and tablets, with focus on mobile phones. The programming language is Java for the logic and XML for layouts. The implementation happened in different steps, each of which is described in the following sections.

4.1 Data model

The first step involved defining in detail what data is needed by the application in order to comply with the requirements and support the necessary functionality. There are multiple entities, including *exercise*, *level*, *level-category*, *geographic object* and *question*, and relations between those. For all this data to be structured, manipulated and queried, a database was necessary. Therefor, an entity-relation model was created which can be seen in Figure 10. A user has multiple exercises, an exercise has multiple level-categories, a level-category has multiple levels and a level has multiple geographic objects. In addition, there is the *quiz* entity which is a singleton and describes an active

quiz. The level-entity holds data about a quiz, whereas the quiz-entity holds data related to a specific quiz-run. A quiz has multiple questions, and each question has one geographic object.

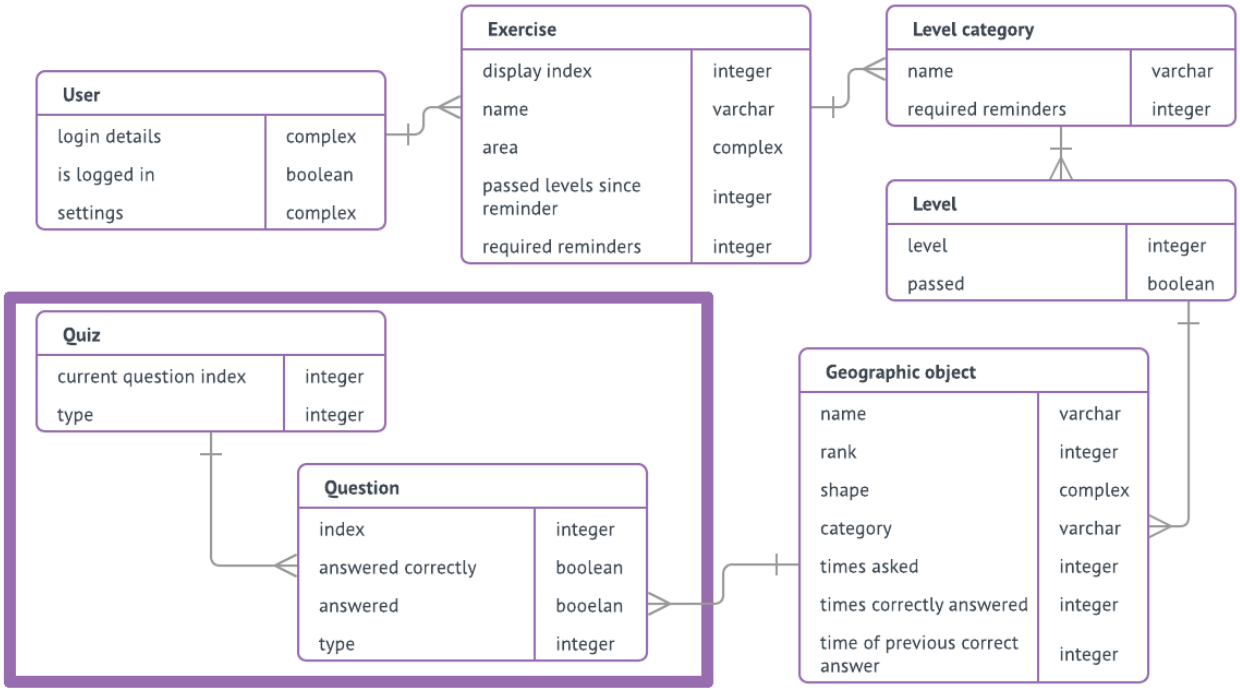

Figure 10: Entity-relation diagram for the permanent data of the application. In addition to stated attributes, each entity has a unique id. The one-to-many relationships are implemented by letting each child reference to the parent through an attribute containing the id of the parent. The purple box contains two entities that are constructed before a quiz. The others entities are more permanent.

The geographic object is the heart of the model. This entity has several attributes:

- **Category** – used during a question in a quiz, where the name along with the category is presented. Examples of values might be *public transport, education, religion, health, food, building, lake, waterfall, village, island and road*. This is a more fine-grained category than the level-category (which is one of *settlement, road, nature, transport* and *construction*).
- **Shape** – defines the shape and location of the object through a set of coordinates. It might be a single node, a line, or a closed line.
- **Rank** – used to divide objects into levels inside a level-category. High rank means the object is of considerable significance and should be placed in an early level, and therefor be presented to the user at an early stage in the exercise progression.
- **Times asked**, **Times correctly answered** and **Time of previous correct answer** – is stats of answers to questions asked about this object. These stats are useful when selecting objects for reminder-quizzes, when determining number of required reminder-quizzes and when selecting geographic objects for a Pair-it question. Stats are not gathered from Pair-it questions, and are of half significance in follow-up quizzes.

Also worth mentioning is the (number of) **Required reminders** attribute of Exercise and Level-category entities. These numbers are semi-randomly updated after each quiz, influenced by the number of problematic objects. A problematic object is an object with a bad success rate, or hasn't been correctly answered in questions for a long time. When these numbers are larger than zero, the level-progression is halted until required reminder-quizzes are completed.

This data-model was implemented using an *SQLite database*. An SQLite engine is available on android devices natively, and useful to store complex data locally on a device.

4.2 Fetching data

Once established what data is needed, it was time to implement the gathering functionality of this data. Most of the data is related to geographic objects. Since the exercise area is arbitrarily specified by the user, and since it's impossible to store detailed geographic data about the whole world in the executable, this data must come from online sources.

One alternative was to create a server which had detailed geographic data stored locally, with world coverage. The server would supply the clients with requested data that would suit the clients' need perfectly. This idea was put aside because of problems that may arise related to server management. Instead, the goal was to let the application download data from public geographic data providers and process this data locally.

It was decided to use geographic data from *OpenStreetMap* (abbreviated *OSM*). OSM is a collaborative project for creating maps of the world, and has extensive coverage which is updated by an active community. OSM-data is free for use by anyone and a number of different APIs exist for extracting the data. (…)

OSM-data consists of elements, which are the basic components of OpenStreetMaps's conceptual model of the physical world. An element is one of three things: Node (a single point in the world defined by longitude and latitude), Way (a collection of nodes in a path) or Relation (a collection of nodes, ways or other relations). In additions to coordinates, an element may have one or many tags. A tag is simply a key and a value; two arbitrary strings. There is no fixed dictionary of of tags but there exists conventions how they are supposed to be used. Since OpenStreetMap is a collaborative project (meaning the data can be added and modified by anyone), the quality and relevance of tagging varies. Some examples of tags are *name=Uppsala* and *place=City*. (https://wiki.openstreetmap.org/wiki/Elements)

Location Educator needs small portions of the OSM-data, specifically geographic elements inside a size-limited polygon. For this purpose, the *Overpass API* service is well suited. This is an API that is optimized for providing OSM-data for a small part of the world. You send a query to the Overpass API using the *Overpass QL* query language and get back the dataset that corresponds to this query. (...)

Using the Overpass QL language, a query was constructed for fetching every element with a name-tag inside a polygon defined by a set of coordinates. Using some special features of the Overpass API it was possible to make Overpass perform some data processing on the raw OSM-data before the data was delivered. Through this, it was possible to receive only the *hull* (an encapsulating path)

around complex objects instead of the whole objects. This shrunk the size of the data downloads considerably and decreased the processing time on the client side.

4.3 Processing data

After the data has been fetched from the Overpass server, it needs to be processed into entities of the data-model. This involves several steps.

4.3.1 Incoming data → build instructions

The incoming data arrives line by line of xml-data(...). A number of lines together describe an OSM-element. It is not preferable to download all data in a big chunk since that might cause the memory to run out on resource-limited mobiles. Therefore, this was implemented in a way that very little data needs to be in main memory at the same time. An incoming line of xml-data is scanned with regular expressions in order to extract the information in the line; Is it a beginning of a new element, a part of the geographic shape of an element, or tags describing an element? This information is collected in a set of build instructions – instructions for building a single geographic object of the data-model. When a line of xml-data contains the beginning of a new geographic element, the set of build instructions currently collected is complete and used to build a geographic object. This construction process is described below. When construction of a geographic object is done, it’s stored in the application database and no longer needs to be stored in main memory.

4.3.2 Build instructions → geographic object

The gathered build instructions for building a geographic object contains a location and shape of a geographic element along with a set of tags describing the element. Only elements with a name-tag are fetched from the Overpass server. Therefore, it’s guaranteed that one of the tags in the tag-set is a name but apart from that, the set of tags can be anything.

For the construction of a geographic object to be possible, a name along with a *super-category* and a *sub-category* is necessary. The super-category corresponds to the broad exercise-category and the sub-category is the more precise category for a particular geographic object. A closed set of super-categories as well as sub-categories was defined, and it is necessary to assign one category from each set to each geographic object during construction. Therefor, the set of tags in the build instructions needs to be converted into two categories. If this conversion fails, the tags aren’t satisfactory and the construction of this particular geographic object isn’t successful.

The conversion from tags into a super- and sub-category isn’t a trivial task. The problem is varying quality of the tags. Conventions exists of how tagging should be done in OSM, but there are no guarantees and you never know what tags will appear. Also, an element usually have multiple tags; how to decide which one to pay attention to?

This was solved by creating a look-up table, listing all tags of interest and defining a super- and sub-category to each. The order of the list defines the importance of a tag, where low index means high importance. This order was roughly estimated. For each tag in the build instructions, the corresponding entry is found in the look-up table, and the super- and sub-category of the tag with lowest index in the table becomes the categories of the geographic object.

In order to create this look-up table for conversion from tags to categories, it was necessary to examine tagging patterns in the OSM-data. For this, TagInfo (…) was used. This tool analyzes all OSM-data regularly, and reveals among other things rankings of tag-popularity and relations between tags.

In addition to a name, a shape, a super-category and a sub-category, a geographic object needs a ranking. This is necessary in order to present the user with significant and *important* (in the sense, important to learn its name and location) geographic objects early, and move on to less important objects later. It's not enough to only use an object's size for this; there may be a huge old warehouse that is of much less interest than a compact town hall. It's not possible to extract a very accurate importance ranking using OSM-data alone. One solution would be to look at statistics from an OSM-element's Wikipedia-page (…) which may provide accurate importance, but it wouldn't be feasible to query for this information for each of a significant amount of geographic object. This was instead solved by extracting an importance-ranking from an OSM-element's number of edits in its version-history, the number of tags present, and finally the element's size (where longer/ bigger means higher importance). This importance-ranking method is questionable but does provides some relevance, and is achieved using OSM-data alone.

4.3.1 Final processing of geographic objects

When all incoming OSM-data has been attended there probably exists multiple new geographic objects in the application database. Once all these objects are present, there is need for some final processing of these objects; some of them need to be merged. The OSM-data contains many elements that aren't complete by themselves, for instance road segments. These are named elements which lie close to another element with same name. This isn't acceptable for the topical purpose; the answer to a question in a quiz may have multiple correct answers, which really isn't preferable. Therefore, merging is attempted by such objects.

All object with same names are extracted as groups. Merge is attempted for objects in each group. A merge between two objects is possible if the objects are in relatively close proximity. These two objects now become one object containing two shapes.

This is a vital step but it takes a long time to complete if there are a lot of geographic object. One reason for this is because it's not preferable to keep all objects in main memory at the same time – the device might run out of memory (a big exercise-area might contain tens of thousands of objects, and an object may contain hundreds of nodes). Therefor, objects need to be loaded from the database repeatedly.

4.3.2 Geographic objects → complete exercise data

When above steps are complete, there are almost complete geographic objects in the database. These objects are all connected to the exercise currently under construction. The quiz-id attribute of the geographic objects are at this point is set to minus one. Now, the complete structure described in the data-model is constructed from the geographic objects as follows:

- Geographic objects are grouped into levels based on super-category and ranking.
- Levels are grouped into level-categories based on the super-category of the objects.

- The level-categories are assigned to a newly constructed exercise.
- The exercise is assigned to a user.

4.4 Common operation

The most complex operation performed by the application is the construction of a new exercise. Once an exercise is created, the application simply presents the data in the database in a relevant way, defined by the application design and requirements.

When a quiz is initiated by the user, a corresponding quiz object is constructed and placed in the database. The database only holds one quiz object at the time, which is removed from the database when a quiz is completed. If a user leaves the application during a quiz, the quiz object stays in the database and when the user returns, the quiz is continued.

During a quiz, stats are gathered from the answers about the geographic objects of the questions. These stats are later used when constructing questions for reminder quizzes, as well as during construction of Pair-it questions. When a quiz is completed by the user, the number of required reminder-quizzes (described in the application design) is determined. This number is semi-randomized and also depends on the geographic objects' stats.

4.5 Integrated maps

During user's specification of a new exercise, during tapping-exercises and during quiz-exercises there is need for an integrated map on the screen. This was implemented using functionality provided by *Mapbox*. The *Mapbox Maps SDK for Android* provides highly customizable integrated maps for the Android platform; all the requirements of integrated maps of Location Educator. The map is customized using a *Mapbox style document* which is a *JSON object* specifying certain design properties.

- **New exercise specification** requires an integrated map with functionality to draw a path. The user is presented with a map that a shape can be drawn on top of, provided the map is zoomed in to a certain zoom level (as defined by the application design). When the user taps the zoomed-in map, a marker is placed on the map at this location. Placed markers are connected with segments. Once three markers are placed, the shape automatically closes with the final segment. More markers can be added to further refine the shape; any number of markers is allowed. The user may clear added markers by tapping a button. If segments are intersecting, this is detected using linear algebra and the *Create exercise button* is disabled. The map presented to the user contains text-labels; text in the map for the names of the various places. The Mapbox Maps SDK allows markers of custom appearance to be placed anywhere on the map. The SDK also provides *Polylines* for representing segments between points.
- **Tapping-exercise** requires an integrated map containing representations of geographic objects. Objects are represented with map annotations; markers and polylines. Polylines are used for both open ended geographic objects (like roads) and objects with a closed shape (like areas of any sort). The SDK provides tapping functionality for these map annotations.

When a user taps an annotation, the corresponding object's name is presented. The underlying map doesn't contain any text-labels. These labels would provide the user with the names of the geographic objects and would contradict the purpose of the exercise. The hiding of text-labels is specified in the Mapbox style document.

- **Quiz-exercise** requires an integrated map containing geographic objects, similarly to the tapping-exercise. This map also doesn't contain any text-labels. Depending on the type of question, geographic objects can be tapped. If a correct answer is provided, the appearance of the object's representation in the map changes to indicate this, which is done by enlarging the markers and thickening the polylines. The color of the geographic objects are initially gray but when tapped turns into the particular color of this object.

4.6 Development tools

For development of the application, Android Studio was used. This is an integrated development environment specifically designed for Android development and provides much functionality that drastically simplifies the production process. Among other things, it provides a layout editor where the developer can create layouts and corresponding XML code using a graphical interface.

Android Studio also provides a version control system. In order to back up the development progress, the version control system *git* was used for storing the code on *github* servers.

5 Evaluation

The implementation of the application was overall successful, and resulted in an application that corresponds well with the functional and non-functional requirements, and the intended design. Some screenshots of the application can be seen in figures 11, 12 and 13.

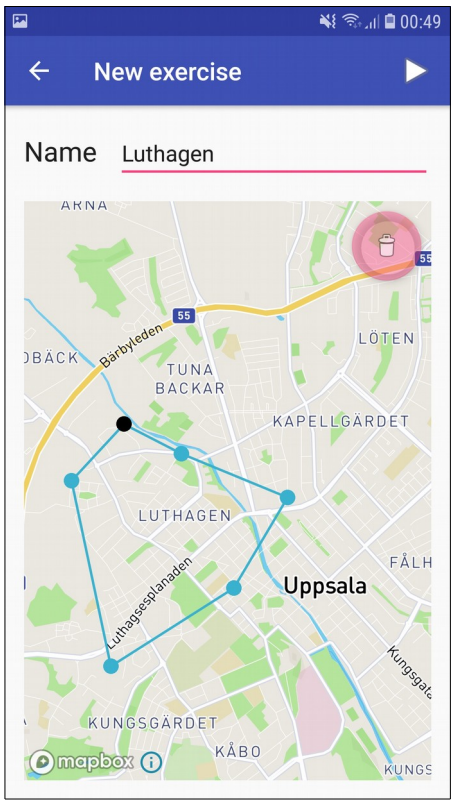

Figure 11: Screenshot of a New exercise activity.

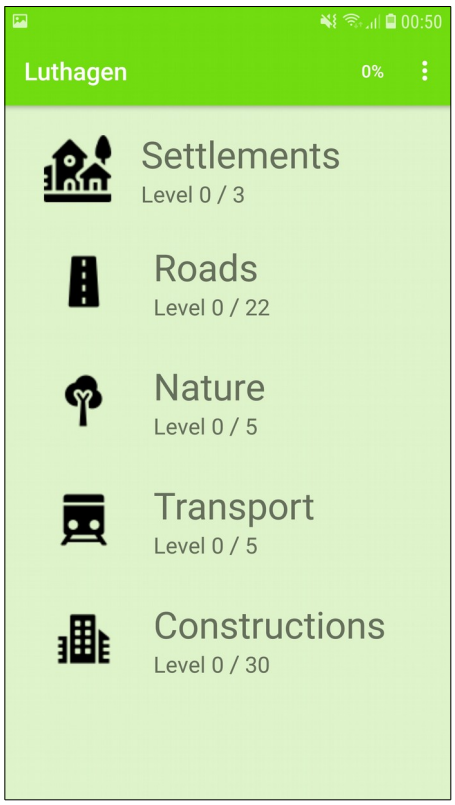

Figure 12: Screenshot of the main activity of an exercise called Luthagen.

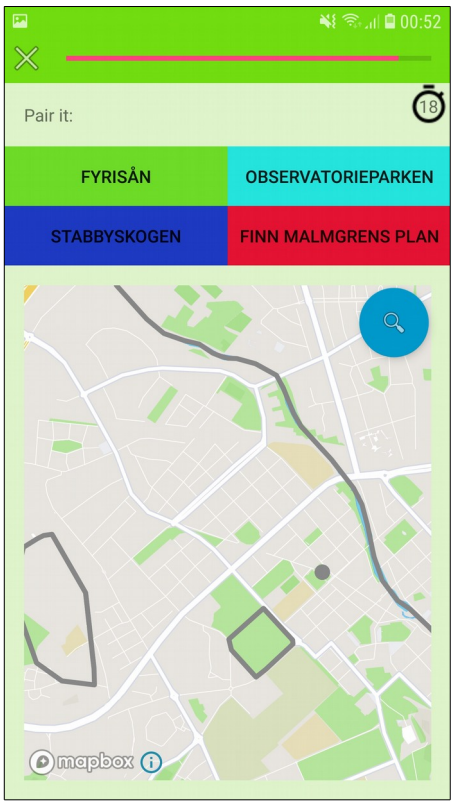

Figure 13: Screenshot of a Pair-it question during a quiz.

5.1 Follow-up on initial user considerations

The finished product was tested by a potential user for an evaluation of the educational value, as a follow-up to the initial user considerations...

6 Current limitations and future work

The functional requirements declare functionality for predefined exercise areas, exercise sharing and progress sharing. None of these are implemented currently. These requirements exist mainly for some user's need of comparing progress with other users, and perhaps monitoring the progress of other users. This is important since some users may find the education more meaningful and motivational if it's put in a social context.

Another limitation of the current implementation is the fact that when the application's data is cleared, like for instance if the application is reinstalled, all exercises and progress is lost. An improvement would involve backing up data on a server, which might not be trivial since an exercise could contain data of a significant size. This would also mean that a user can switch device and still keep the exercises and progress.

Another weakness in the current implementation is the inaccurate estimate of the importance of a geographic object (used to determine a ranking among them for grouping them into levels). A solution may involve looking at statistics from an OSM-element's Wikipedia-page. It's not possible for a device to query for this information for each geographic object since it would take too much

time. A server would be necessary, where relevant data for the whole planet is obtained and stored locally. The server would then supply the clients with the precise data a client needs. This could also provide a performance increase in exercise creation since the processing step currently performed on the device wouldn't be necessary. However, this update involves much work and requires the need of one or many servers. The benefits seems relatively small compared to the effort.

Finally, the merging-step during creation of new geographic objects suffers from bad performance, which definitely could be improved through optimizations of the current implementation.

7 Discussion

The reason I wanted to develop this application was because I found myself looking for an existing application of this sort but couldn't find one. I was using an Android application for language studies and realized the educational power of this. A friend of mine even learned to speak Swedish primarily through his tablet. I would like to become better at local geography in the city where I live, the areas in the archipelago of Stockholm where I spend the summers, and generally for places I'm traveling to. The names of local places tell a story about this areas' history. And the names help creating and preserving memories from these places.

Another way to study local geography is by inspecting a map. This might not be ideal for everyone since some might not find it entertaining or very easy to find a way to preserve the names and locations in the memory. But there is a potential comfort in having a physical map to study; the map doesn't change or disappear. You can put it in a drawer and take it out whenever you want to refresh your knowledge. You can trust it in a way a mobile device can't be trusted. The mobile can break, one day it just won't function and all the data is lost. The current implementation of Location Educator doesn't provide any support to battle this issue. When the application's local data is cleared or lost, all exercises and progress is lost. Lack of functionality for backing up the application's data is a severe limitation of the current implementation.

Since the application has been finished I have used it myself for studying the local geography of the city where I'm about to move. I really found the application useful; I learned local geography though it and I did find it entertaining to use. Therefor, I must confess I'm very happy with the result of this project.

References

Vancouver style referencing (using brackets [1]). (Mendeley plugin)

[1] Title, writer, publisher, year...